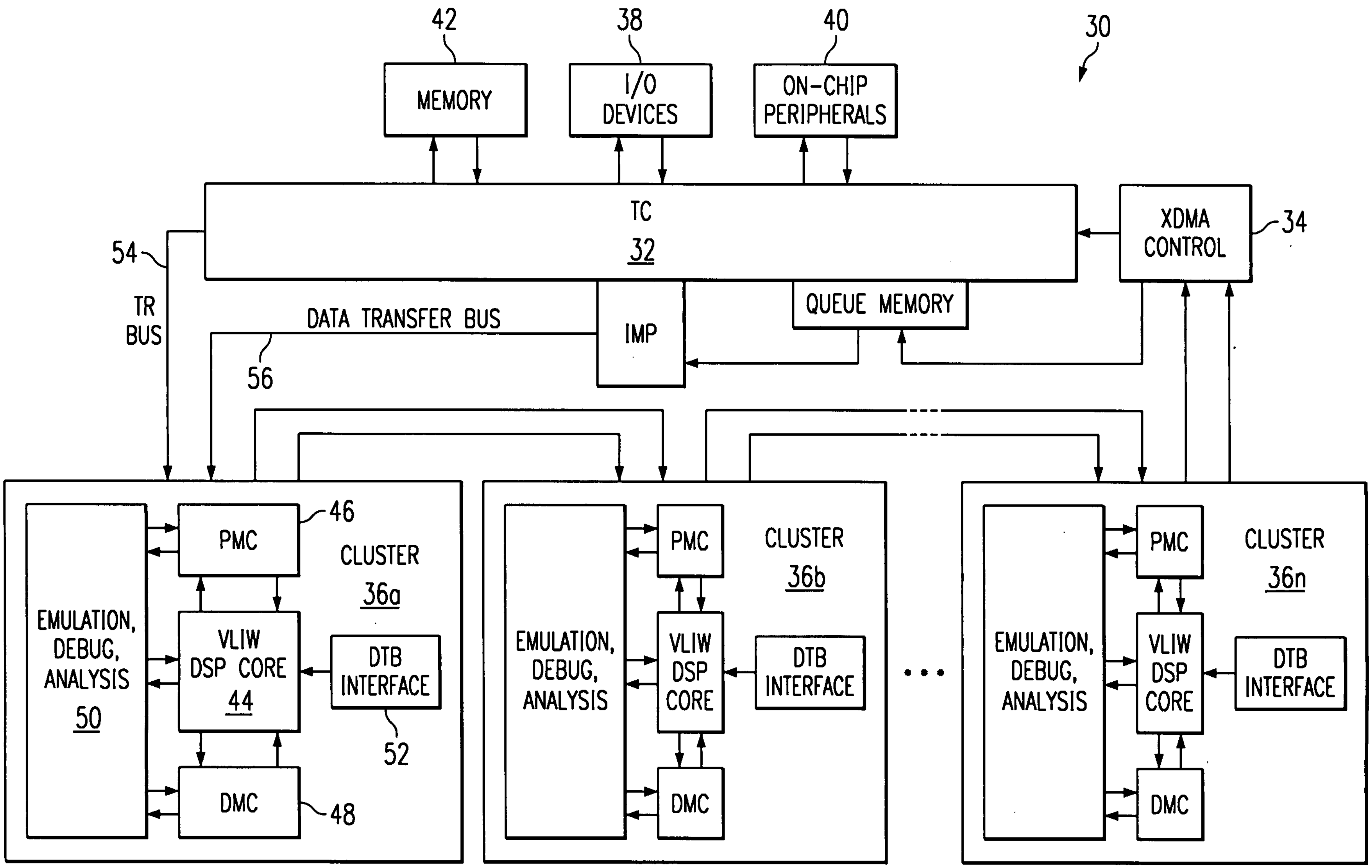
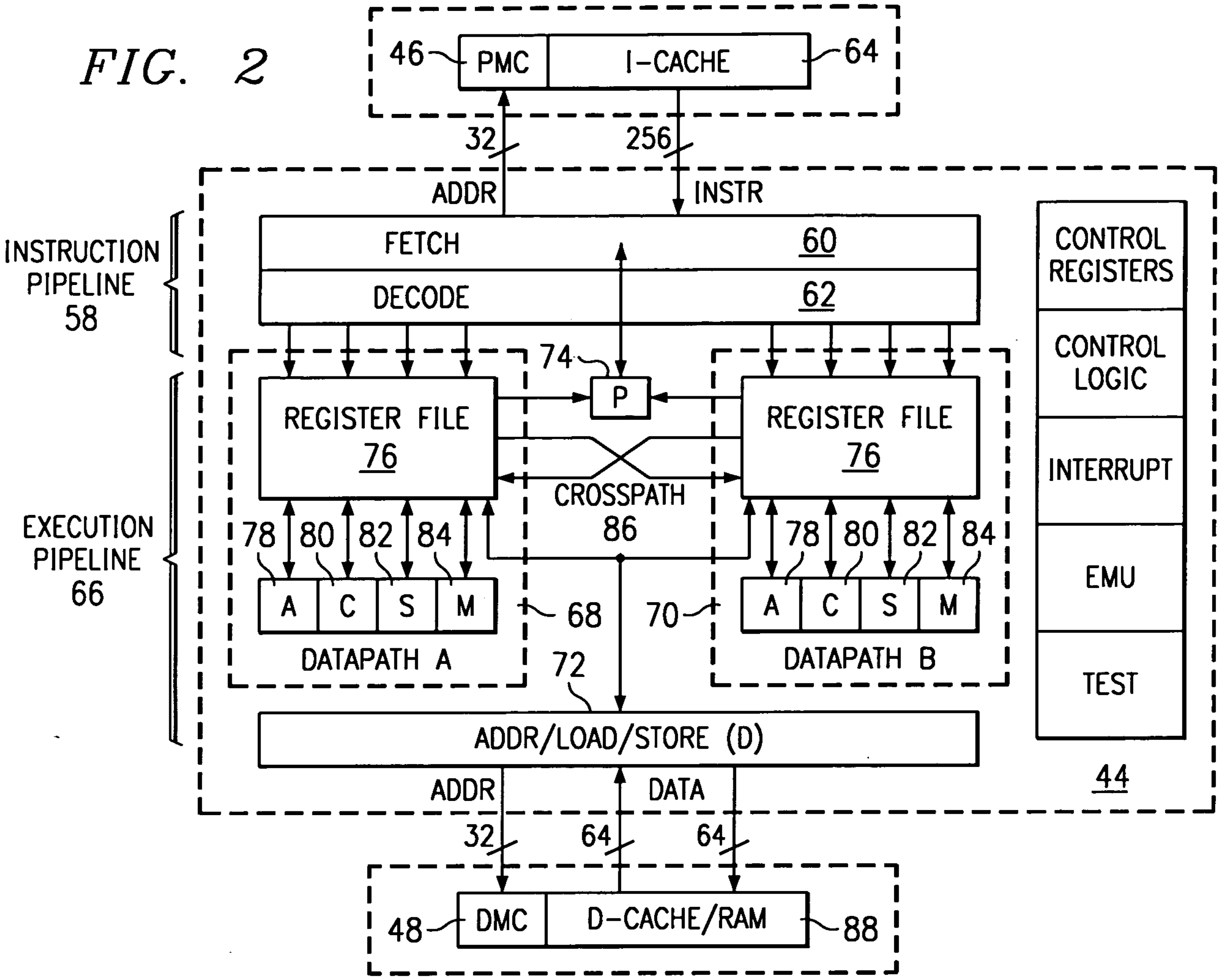

42
MEMORY
38
I/O DEVICES
40
ON-CHIP PERIPHERALS
30
TC
32
XDMA CONTROL
34
54
TR BUS
DATA TRANSFER BUS
56
IMP
QUEUE MEMORY
EMULATION, DEBUG, ANALYSIS
50
PMC
46
VLIW DSP CORE
44
DMC
48
CLUSTER
36a
DTB INTERFACE
52
EMULATION, DEBUG, ANALYSIS
PMC
VLIW DSP CORE
DMC
CLUSTER
36b
DTB INTERFACE
EMULATION, DEBUG, ANALYSIS
PMC
VLIW DSP CORE
DMC
CLUSTER
36n
DTB INTERFACE

FIG. 1

FIG. 2



UNIT GROUP	OPERATIONS	REGISTER FILE ACCESS	
		PRIMARY DATAPATH	ALTERNATIVE DATAPATH
A	- GENERAL ARITHMETIC - BOOLEAN AND CONTROL REGISTER ACCESS	R/W	R
C	- COMPARE, SHIFT, BOOLEAN - ARITHMETIC: ADD, SUB	R/W	R
S	- SHIFT, ROTATE, EXTENDED BOOLEAN - ARITHMETIC: ADD, SUB	R/W	R
M	- MULTIPLY - ARTHMETIC: ADD, SUB	R/W	R
D	- LOAD - STORE - ADDRESS COMPUTATION	W TO BOTH R FROM BOTH R/W BOTH	
P	- BRANCH	R FROM BOTH	

FIG. 3

R=READ, W=WRITE

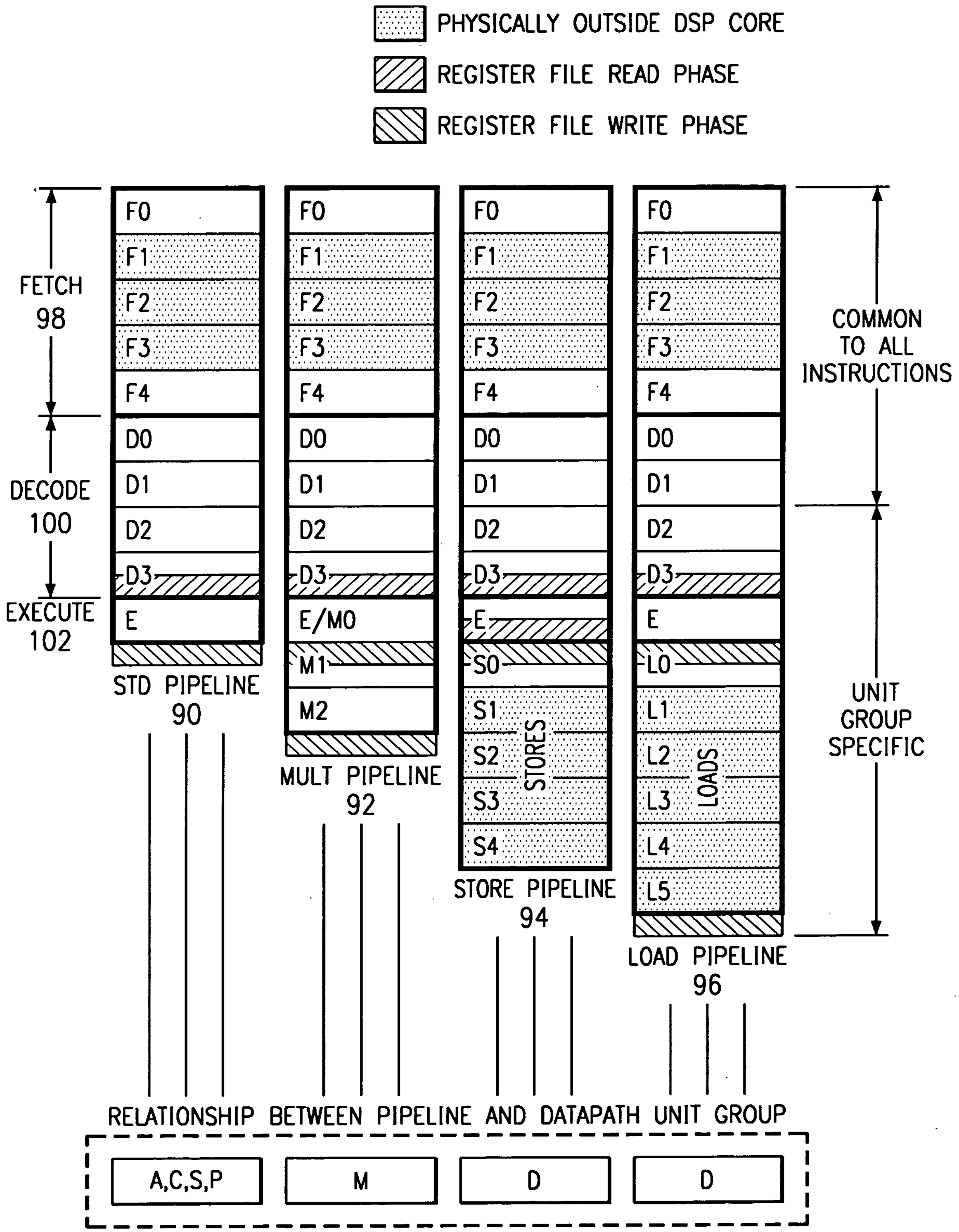

PHYSICALLY OUTSIDE DSP CORE
REGISTER FILE READ PHASE
REGISTER FILE WRITE PHASE
FETCH 98
DECODE 100
EXECUTE 102
F0
F1
F2
F3
F4
D0
D1
D2
D3
E
STD PIPELINE 90
E/M0
M1
M2
MULT PIPELINE 92
S0
S1
S2
S3
S4
STORES
STORE PIPELINE 94
L0
L1
L2
L3
L4
L5
LOADS
LOAD PIPELINE 96
COMMON TO ALL INSTRUCTIONS
UNIT GROUP SPECIFIC
RELATIONSHIP BETWEEN PIPELINE AND DATAPATH UNIT GROUP
A,C,S,P
M
D
D

FIG. 4

STAGE	FUNCTION
F0	SEND PC TO PROGRAM MEMORY CONTROLLER. LDIP ASSIGNED.
F1	CACHE BLOCK SELECT.
F2	ADDRESS PHASE OF INSTRUCTION CACHE ACCESS.
F3	DATA PHASE OF INSTRUCTION CACHE ACCESS.
F4	FETCH PACKET SENT TO DSP.

FIG. 5a

STAGE	FUNCTION
D0	DETERMINE VALID INSTRUCTIONS IN CURRENT FETCH PACKET.
D1	SORTS INSTRUCTIONS IN EXECUTE PACKET ACCORDING TO DESTINATION UNITS.
D2	INSTRUCTIONS SENT TO DESTINATION UNITS. CROSSPATH REGISTER READS OCCUR.
D3	UNITS DECODE INSTRUCTIONS. REGISTER FILE READ (2ND PHASE).

FIG. 5b

UNIT	STAGE	FUNCTION
NON M UNIT	E	EXECUTION OF OPERATION BEGINS AND COMPLETES. FULL RESULT AVAILABLE AT END OF CYCLE.
M UNIT	M0	EXECUTION OF MULTIPLY OPERATION BEGINS. (OR, NON-MULTIPLY OPERATION BEGINS AND COMPLETES.)
M UNIT	M1	MULTIPLY OPERATION CONTINUES. (OR, NON-MULTIPLY RESULT WRITTEN TO REGISTER FILE (PHASE 1).)
M UNIT	M2	MULTIPLY OPERATION COMPLETES.

FIG. 5c

STAGE	FUNCTION
E	ADDRESS GENERATION OCCURS. REGISTER FILE ACCESS FOR READ DATA.
L0	LOAD ADDRESS GENERATED DURING E IS SENT TOWARDS THE DMC.
L1	ADDRESS DECODE, TC ARBITRATION, TAG COMPARES.
L2	ADDRESS DECODE, TC ARBITRATION, TAG COMPARES.
L3	ADDRESS PHASE OF DATA CACHE ACCESS.
L4	DATA PHASE OF DATA CACHE ACCESS.
L5	64-BIT DATA SENT TO DSP.

FIG. 5d

STAGE	FUNCTION
E	ADDRESS GENERATION OCCURS. REGISTER FILE ACCESS FOR WRITE DATA.
S0	ADDRESS SENT TO DMC.
S1	ADDRESS DECODE IN DMC. WRITE DATA ALIGNMENT.
S2	TAG COMPARE IN DMC. WRITE DATA SENT TO DMC.
S3	ADDRESS PHASE IN DATA CACHE.
S4	DATA PHASE IN DATA CACHE.

FIG. 5e

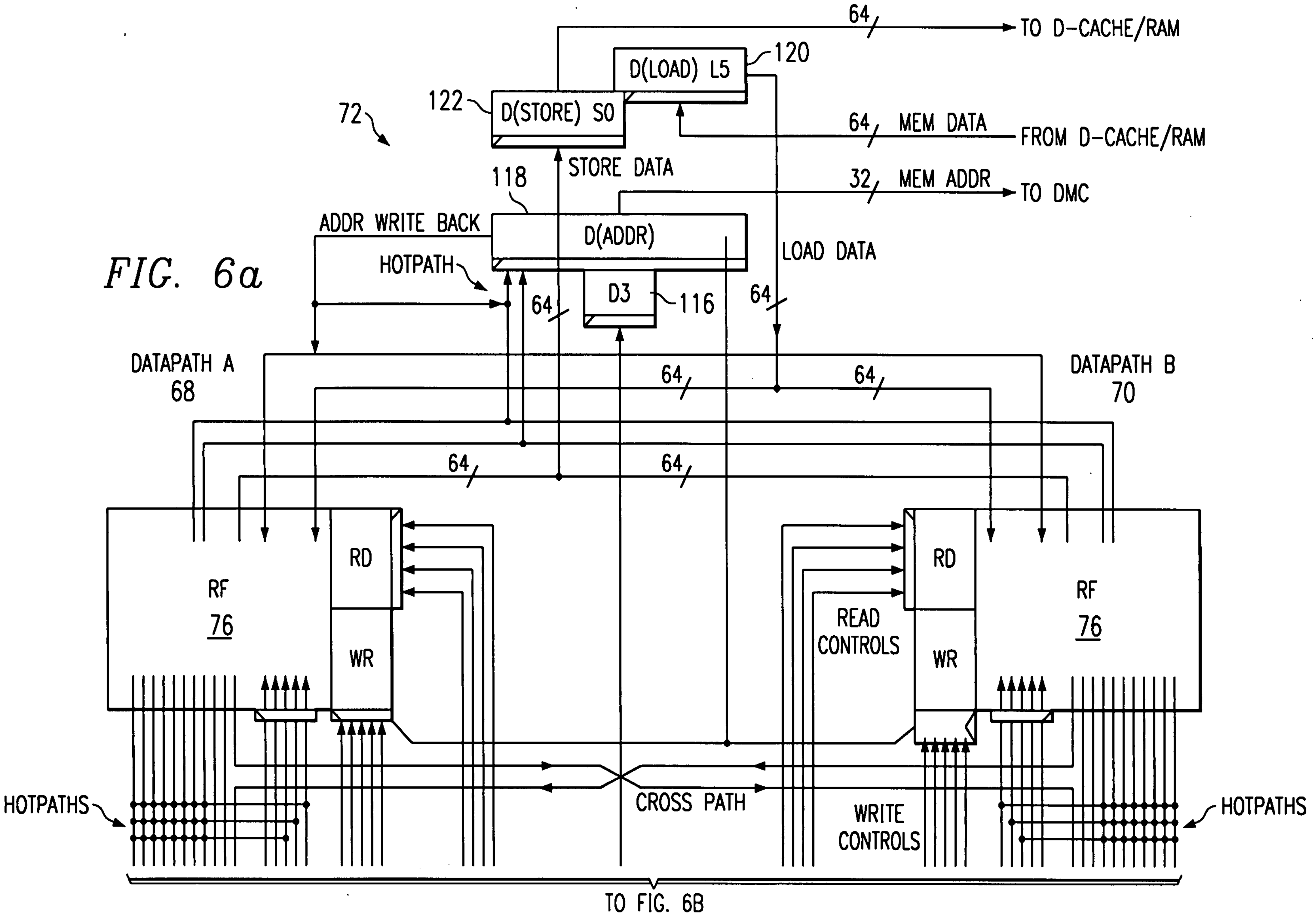

64
TO D-CACHE/RAM
D(LOAD) L5
120
122
D(STORE) S0
72
64 MEM DATA
FROM D-CACHE/RAM
STORE DATA
118
32 MEM ADDR
TO DMC
ADDR WRITE BACK
D(ADDR)
LOAD DATA
HOTPATH
64
D3
116
64
DATAPATH A
68
64
64
DATAPATH B
70
64
64
RD
RF
76
WR
RD
READ CONTROLS
WR
RF
76
HOTPATHS
CROSS PATH
WRITE CONTROLS
HOTPATHS
TO FIG. 6B

FIG. 6a

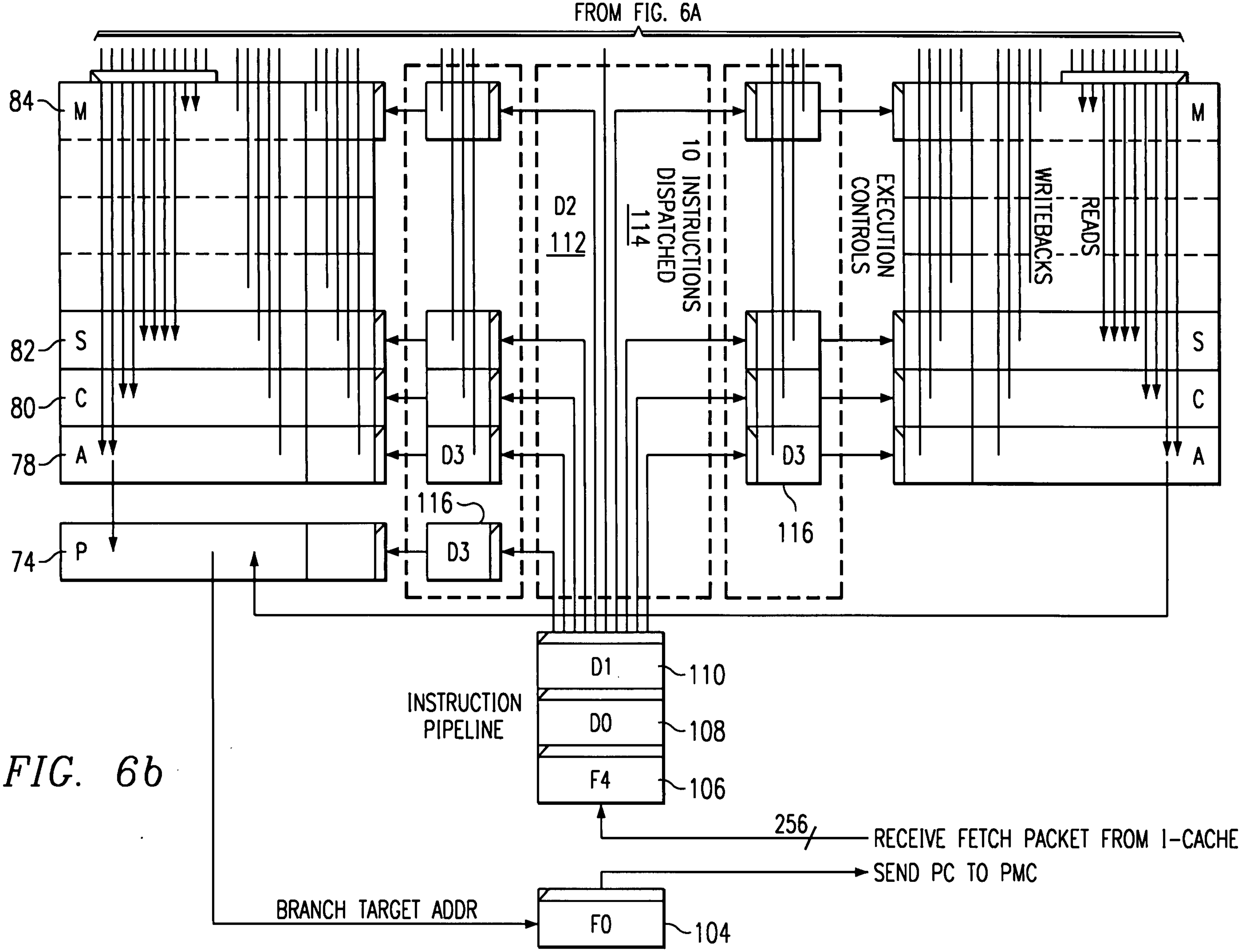

FROM FIG. 6A
84
M
82
S
80
C
78
A
74
P
116
D3
D3
D2
112
10 INSTRUCTIONS
DISPATCHED
114
D3
116
EXECUTION
CONTROLS
WRITEBACKS
READS
M
S
C
A
D1
110
D0
108
F4
106
INSTRUCTION
PIPELINE
256
RECEIVE FETCH PACKET FROM I-CACHE
SEND PC TO PMC
BRANCH TARGET ADDR
F0
104

FIG. 6b

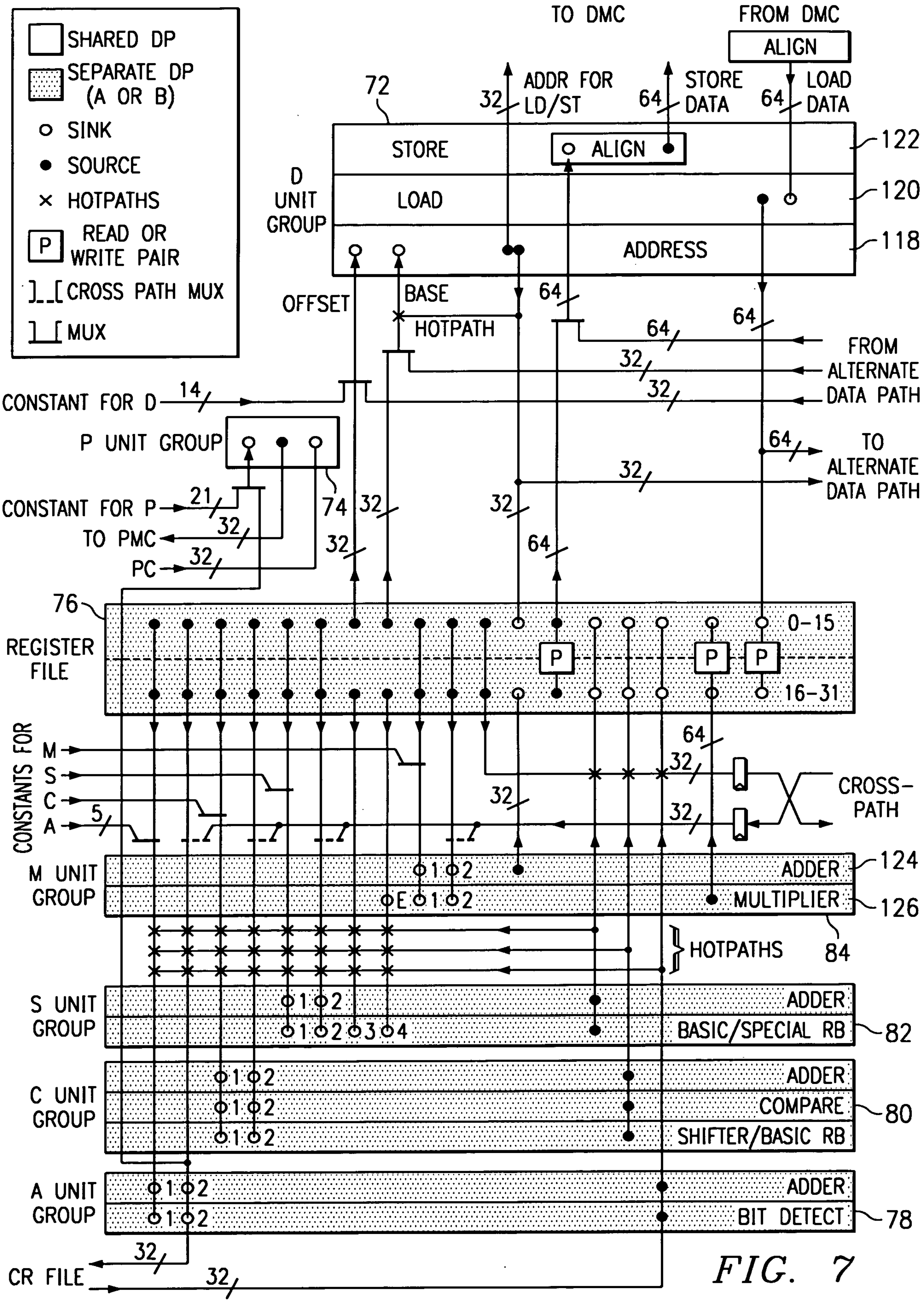

SHARED DP
SEPARATE DP (A OR B)
SINK
SOURCE
× HOTPATHS
P READ OR WRITE PAIR
CROSS PATH MUX
MUX
TO DMC
FROM DMC
ALIGN
ADDR FOR LD/ST
STORE DATA
LOAD DATA
32
64
64
72
STORE
ALIGN
LOAD
ADDRESS
122
120
118
D UNIT GROUP
OFFSET
BASE
HOTPATH
64
64
64
32
32
FROM ALTERNATE DATA PATH
CONSTANT FOR D
14
P UNIT GROUP
64
TO ALTERNATE DATA PATH
32
CONSTANT FOR P
21
TO PMC
32
PC
32
74
32
32
32
64
76
REGISTER FILE
0-15
16-31
P
P
P
CONSTANTS FOR
M
S
C
A
5
64
32
32
32
CROSS-PATH
M UNIT GROUP
1 2
E 1 2
ADDER
MULTIPLIER
124
126
84
HOTPATHS
S UNIT GROUP
1 2
1 2 3 4
ADDER
BASIC/SPECIAL RB
82
C UNIT GROUP
1 2
1 2
1 2
ADDER
COMPARE
SHIFTER/BASIC RB
80
A UNIT GROUP
1 2
1 2
ADDER
BIT DETECT
78
CR FILE
32
32

FIG. 7

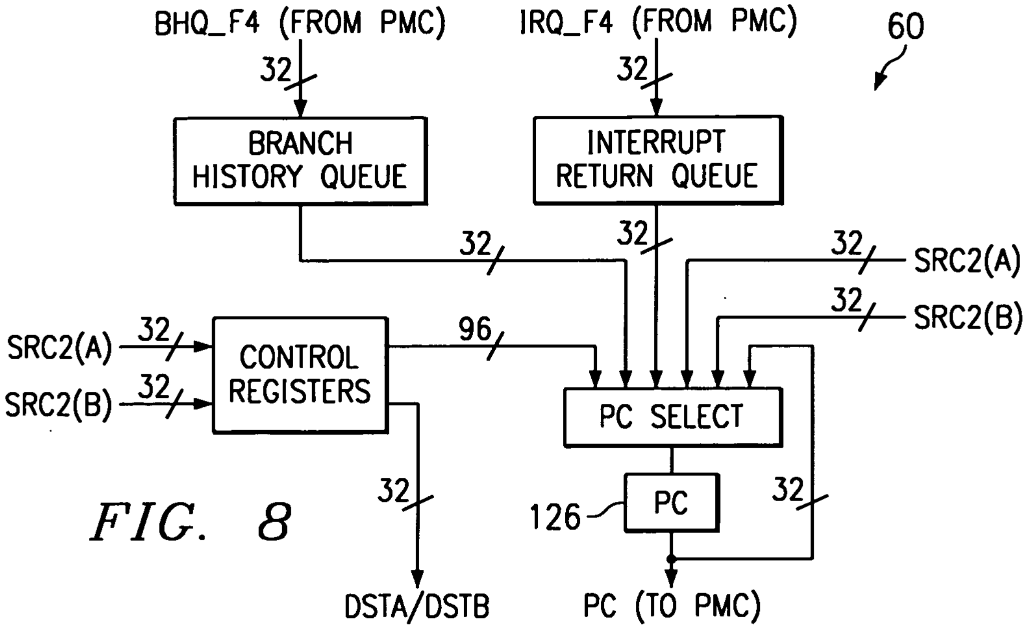


FIG. 8

|| [PREDICATION REG] INSTRUCTION_MNEMONIC .UNIT-DATAPATH-CROSSPATH OP1, OP2, DST

WHERE:

||	=TO BE SCHEDULED IN PARALLEL WITH PRECEDING INSTRUCTION(S)
[PREDICATION REG]	=REGISTER CONTAINING PREDICATION VALUE
.UNIT	=A,C,S,M,D,P UNIT GROUPS
DATAPATH	=1 FOR DATAPATH A, 2 FOR DATAPATH B
CROSSPATH	=X IF ONE OPERAND COMES FROM OPPOSITE REGISTER FILE
OP1, OP2	=SOURCE REGISTERS
DST	=DESTINATION REGISTER

UNIT GROUP	ASSEMBLY NOTATIONS		ASSEMBLY EXAMPLES	WITH CROSSPATH
	DATAPATH A	DATAPATH B		
A	.A1	.A2	ADD .A1 A1,A2,A3 SUB .A2 B1,B2,B3	ADD .A1X A1,B2,A3 SUB .A2X B1,A2,B3
C	.C1	.C2	CMPEQ .C1 A1,A2,A3 CMPEQ .C2 B1,B2,B3	CMPEQ .C1X A1,B2,A3 CMPEQ .C2X B1,A2,B3
S	.S1	.S2	SHL .S1 A1,A2,A3 SHL .S2 B1,B2,B3	SHL .S1X A1,B2,A3 SHL .S2X B1,A2,B3
M	.M1	.M2	MPY .M1 A1,A2,A3 MPY .M2 B1,B2,B3	MPY .M1X A1,B2,A3 MPY .M2X B1,A2,B3
D	.D		LDB .D *A8,A12 STB .D A8,*A12 ADDAH .D A8,A2,B1	n/a
P	.P		B A8	n/a

FIG. 15

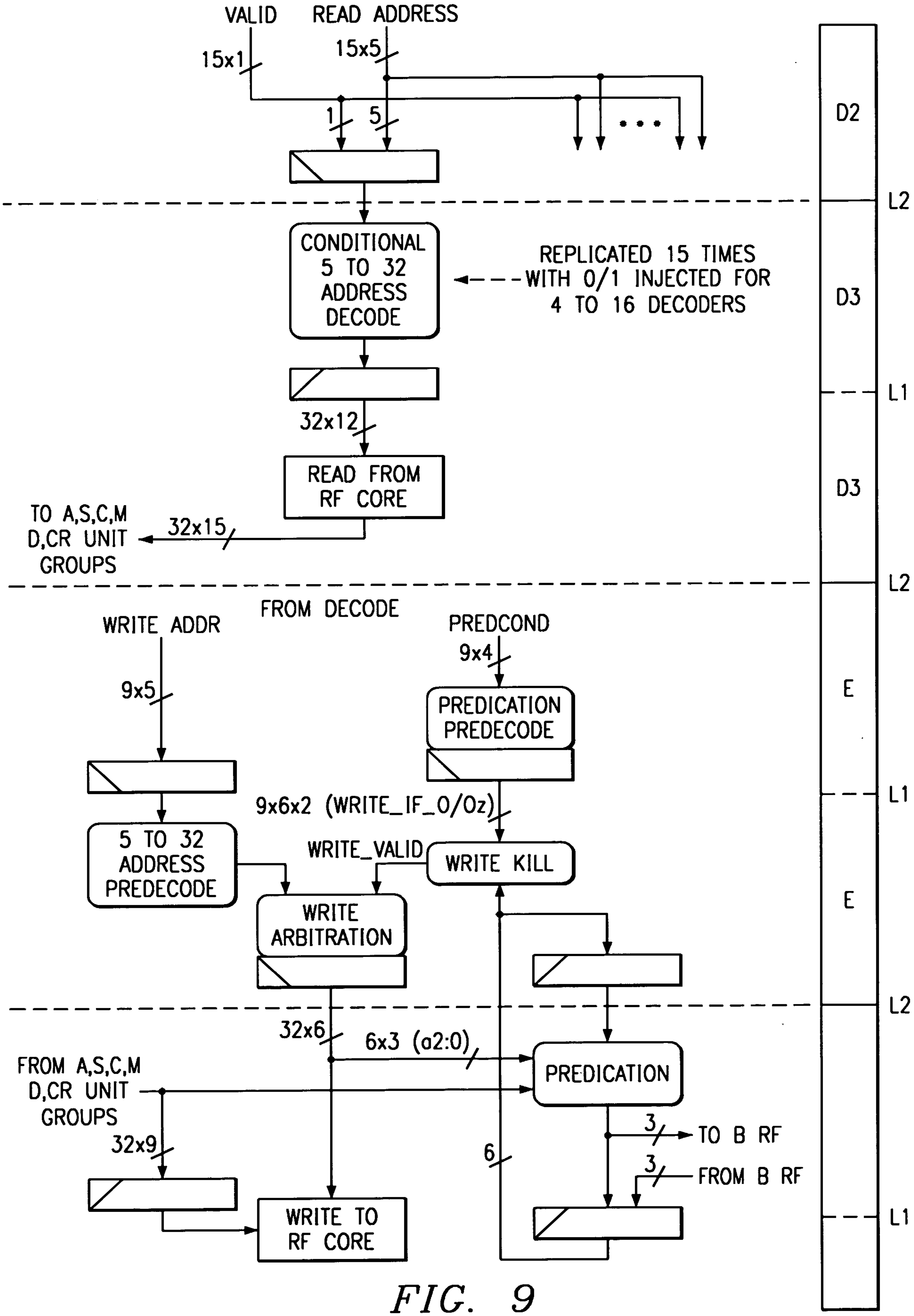

VALID
READ ADDRESS
15x1
15x5
1
5
• • •
CONDITIONAL 5 TO 32 ADDRESS DECODE
REPLICATED 15 TIMES WITH 0/1 INJECTED FOR 4 TO 16 DECODERS
32x12
READ FROM RF CORE
TO A,S,C,M D,CR UNIT GROUPS
32x15
FROM DECODE
WRITE ADDR
PREDCOND
9x4
9x5
PREDICATION PREDECODE
9x6x2 (WRITE_IF_0/0z)
5 TO 32 ADDRESS PREDECODE
WRITE_VALID
WRITE KILL
WRITE ARBITRATION
32x6
6x3 (a2:0)
PREDICATION
FROM A,S,C,M D,CR UNIT GROUPS
32x9
3
TO B RF
6
3
FROM B RF
WRITE TO RF CORE
D2
L2
D3
L1
D3
L2
E
L1
E
L2
L1

FIG. 9

FIG. 10

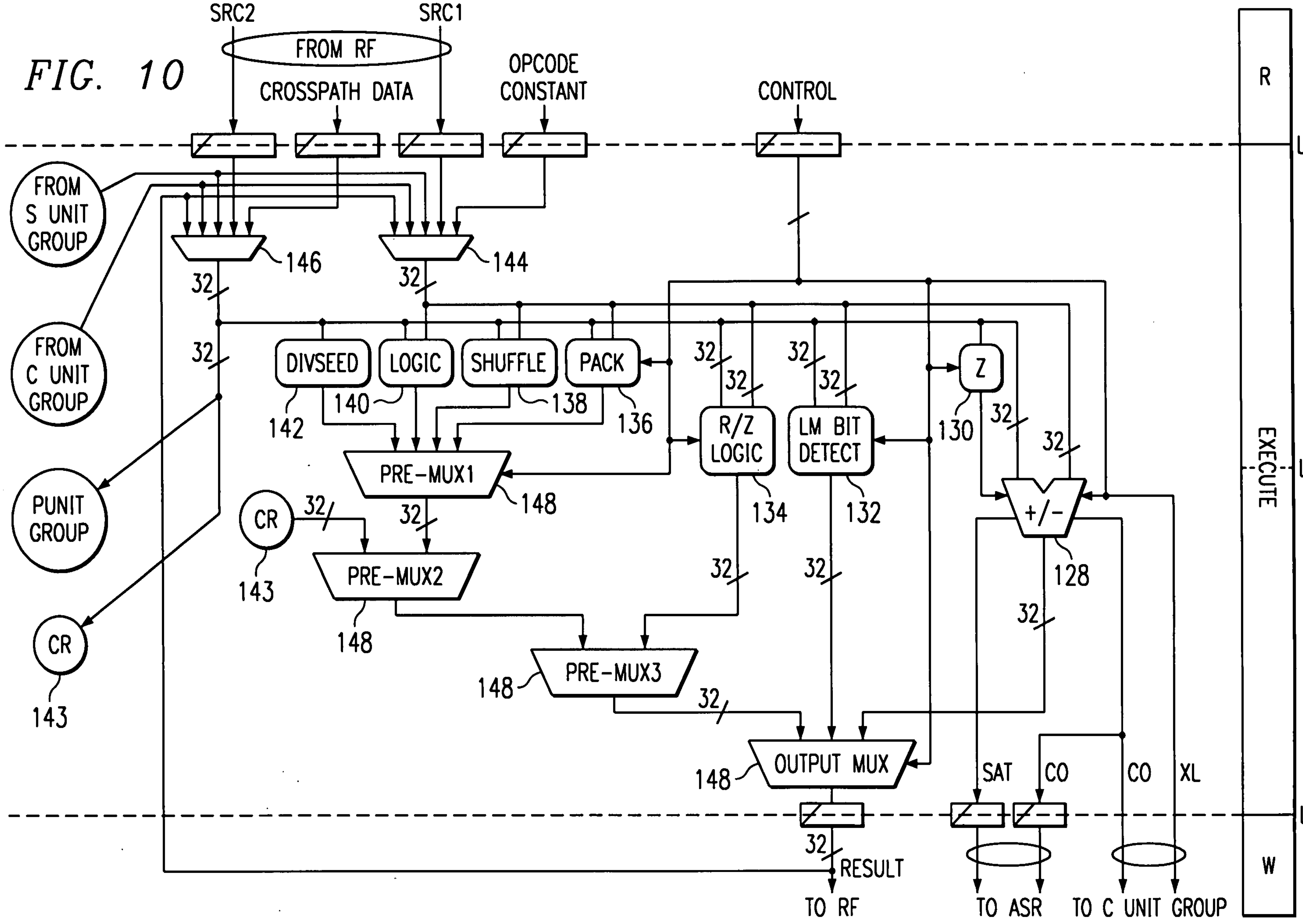
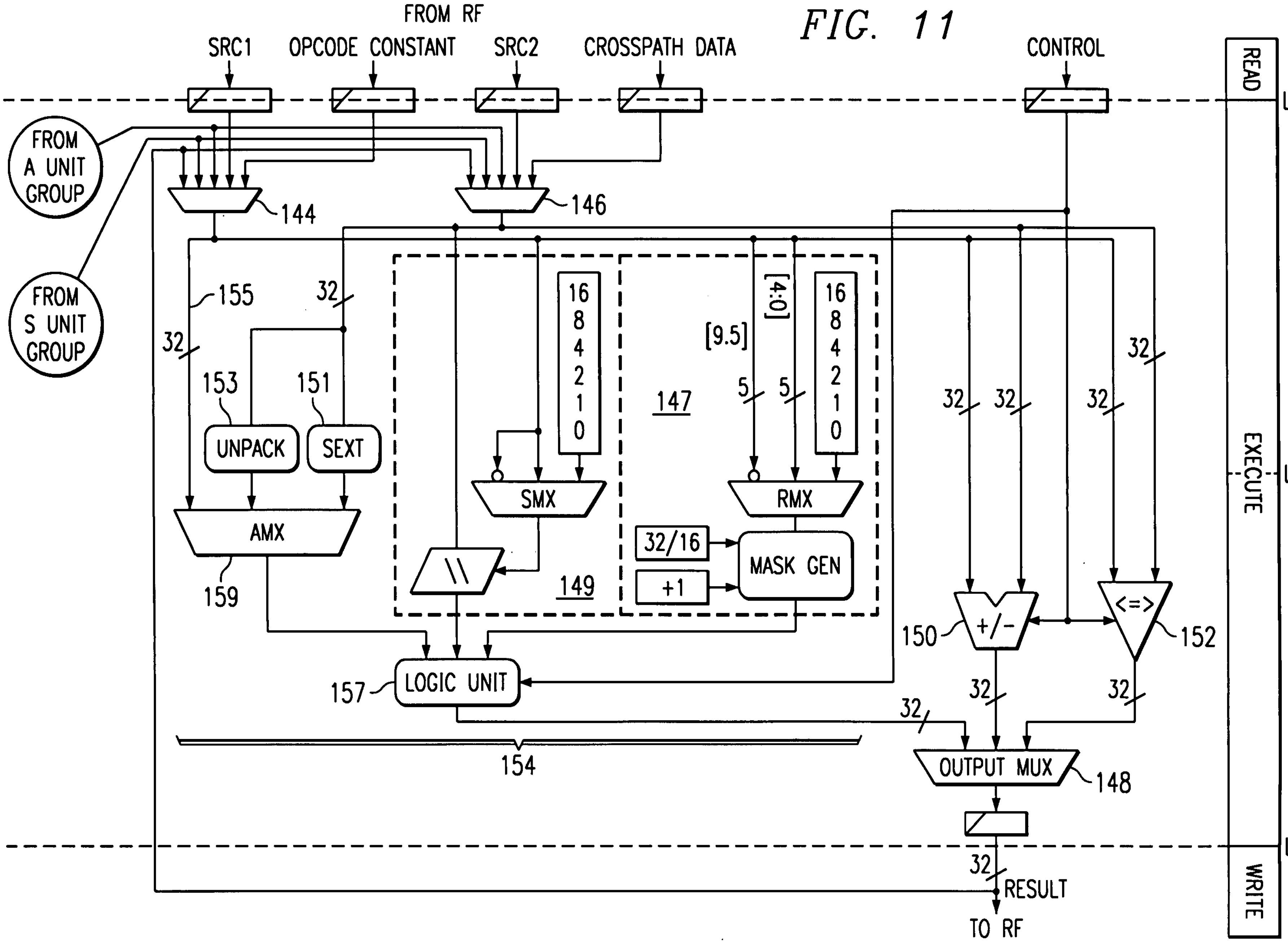

SRC2
SRC1
FROM RF
CROSSPATH DATA
OPCODE
CONSTANT
CONTROL
R
EXECUTE
W
L2
L1
L2
FROM
S UNIT
GROUP
FROM
C UNIT
GROUP
PUNIT
GROUP
CR
143
146
144
32
32
32
DIVSEED
LOGIC
SHUFFLE
PACK
142
140
138
136
32
32
32
32
R/Z
LOGIC
LM BIT
DETECT
134
132
Z
130
32
32
+/-
128
PRE-MUX1
148
CR
32
32
143
PRE-MUX2
148
PRE-MUX3
148
32
32
32
32
OUTPUT MUX
148
SAT
CO
CO
XL
32
RESULT
TO RF
TO ASR
TO C UNIT GROUP
CR
143

FIG. 11


FROM RF
SRC1
OPCODE CONSTANT
SRC2
CROSSPATH DATA
CONTROL
READ
EXECUTE
WRITE
L2
L1
L2
FROM A UNIT GROUP
FROM S UNIT GROUP
144
146
155
32
32
153
151
UNPACK
SEXT
AMX
159
16
8
4
2
1
0
SMX
\\
149
147
[9.5]
[4:0]
5
5
16
8
4
2
1
0
RMX
32/16
+1
MASK GEN
32
32
32
32
+/-
<=>
150
152
LOGIC UNIT
157
32
32
32
154
OUTPUT MUX
148
32
RESULT
TO RF

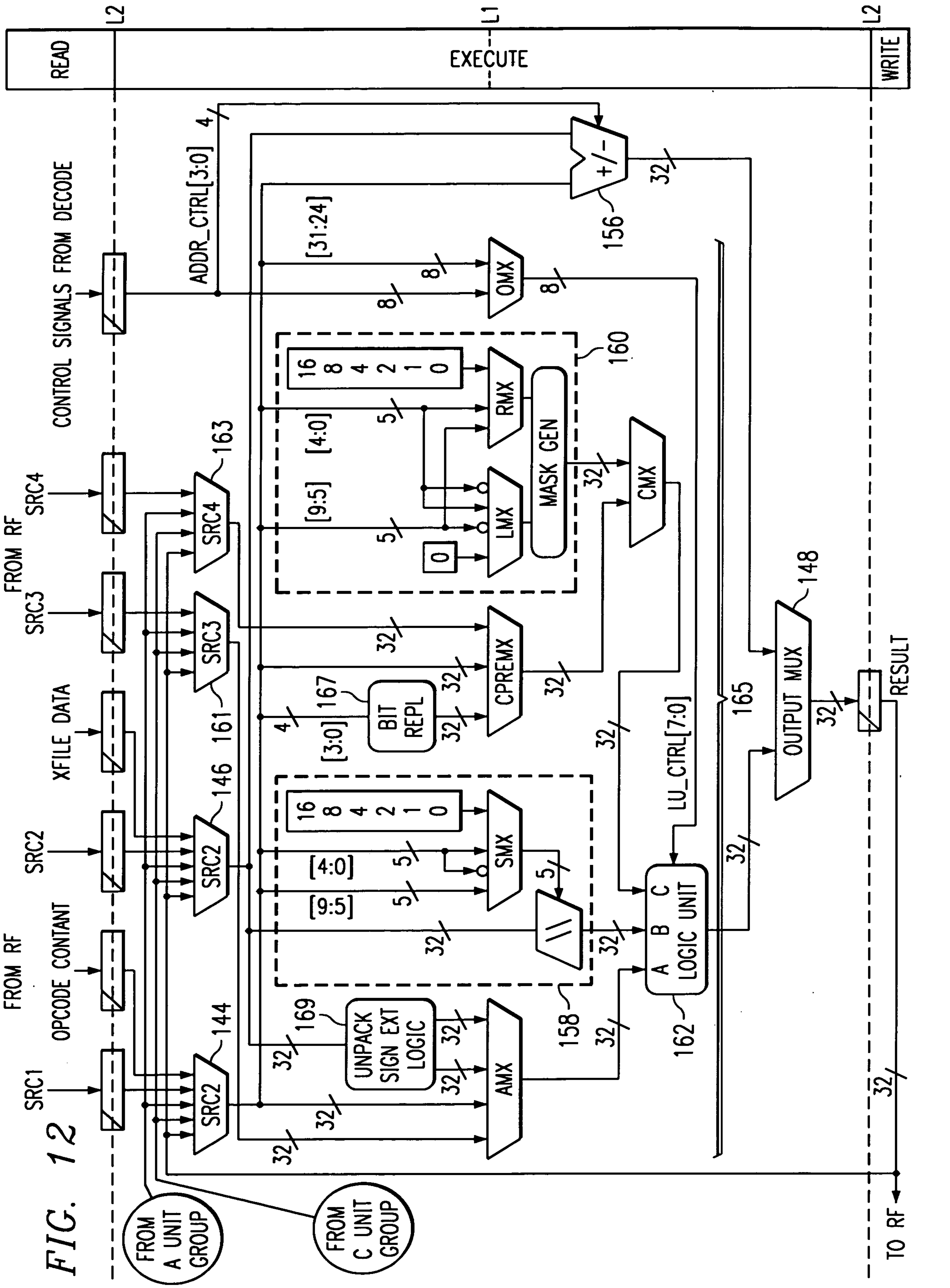

READ
EXECUTE
WRITE
L2
L1
L2
FROM RF
SRC1
OPCODE CONTANT
SRC2
XFILE DATA
SRC3
FROM RF
SRC4
CONTROL SIGNALS FROM DECODE
ADDR_CTRL[3:0]
4
+/-
156
32
[31:24]
8
8
8
OMX
8
160
16
8
4
2
1
0
RMX
[4:0]
5
[9:5]
5
LMX
0
MASK GEN
32
CMX
163
SRC4
SRC3
161
32
CPREMX
32
32
167
BIT REPL
32
4
[3:0]
32
165
LU_CTRL[7:0]
OUTPUT MUX
148
RESULT
32
146
SRC2
16
8
4
2
1
0
SMX
[4:0]
5
5
[9:5]
5
32
32
A B C
LOGIC UNIT
162
32
32
144
SRC2
169
UNPACK SIGN EXT LOGIC
32
32
32
AMX
158
32
32
32
32
FROM A UNIT GROUP
FROM C UNIT GROUP
TO RF

FIG. 12

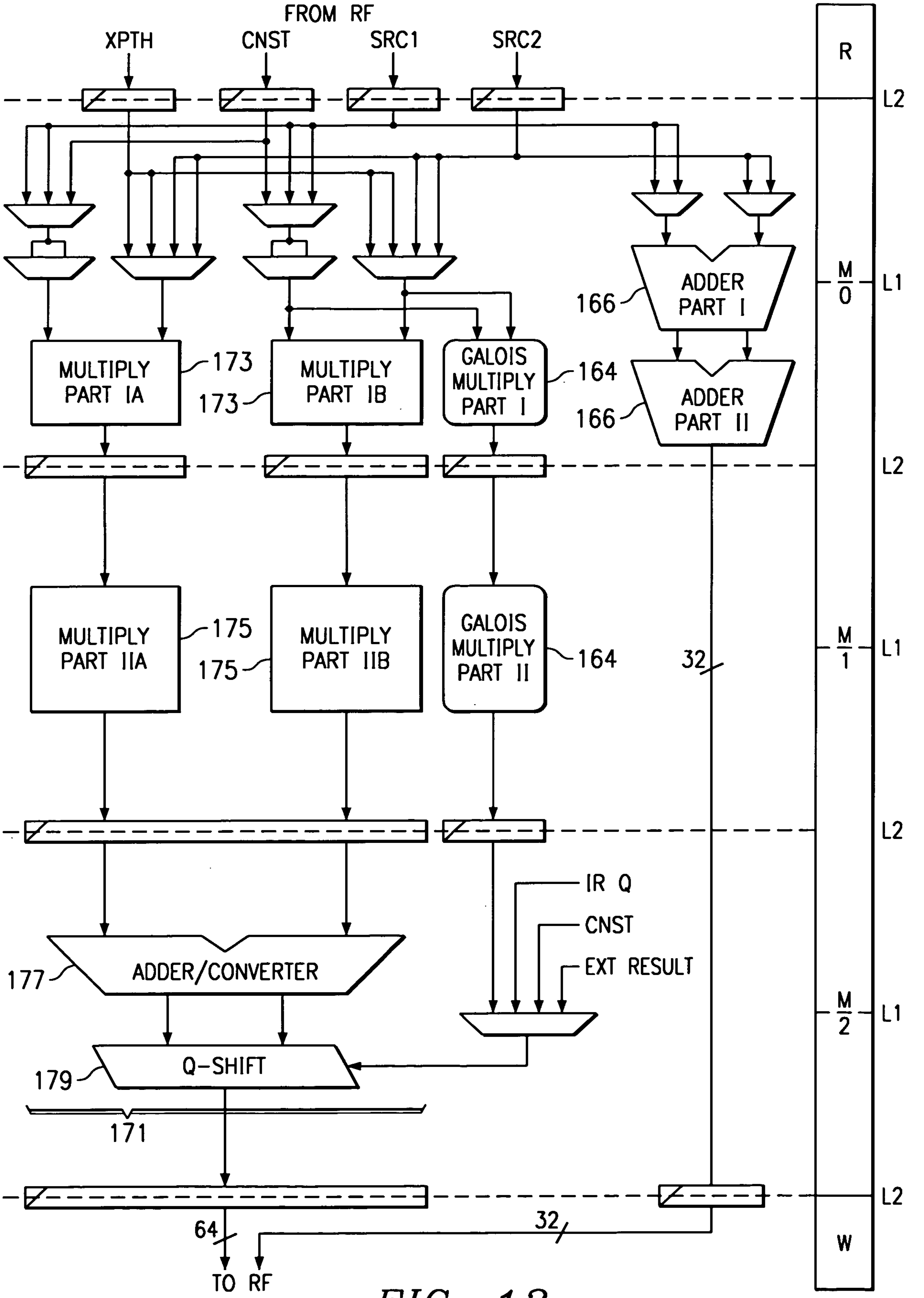

FROM RF
XPTH
CNST
SRC1
SRC2
R
L2
ADDER PART I
166
M 0
L1
MULTIPLY PART IA
173
MULTIPLY PART IB
173
GALOIS MULTIPLY PART I
164
ADDER PART II
166
L2
MULTIPLY PART IIA
175
MULTIPLY PART IIB
175
GALOIS MULTIPLY PART II
164
32
M 1
L1
L2
IR Q
CNST
EXT RESULT
ADDER/CONVERTER
177
M 2
L1
Q-SHIFT
179
171
L2
64
32
W
TO RF

FIG. 13

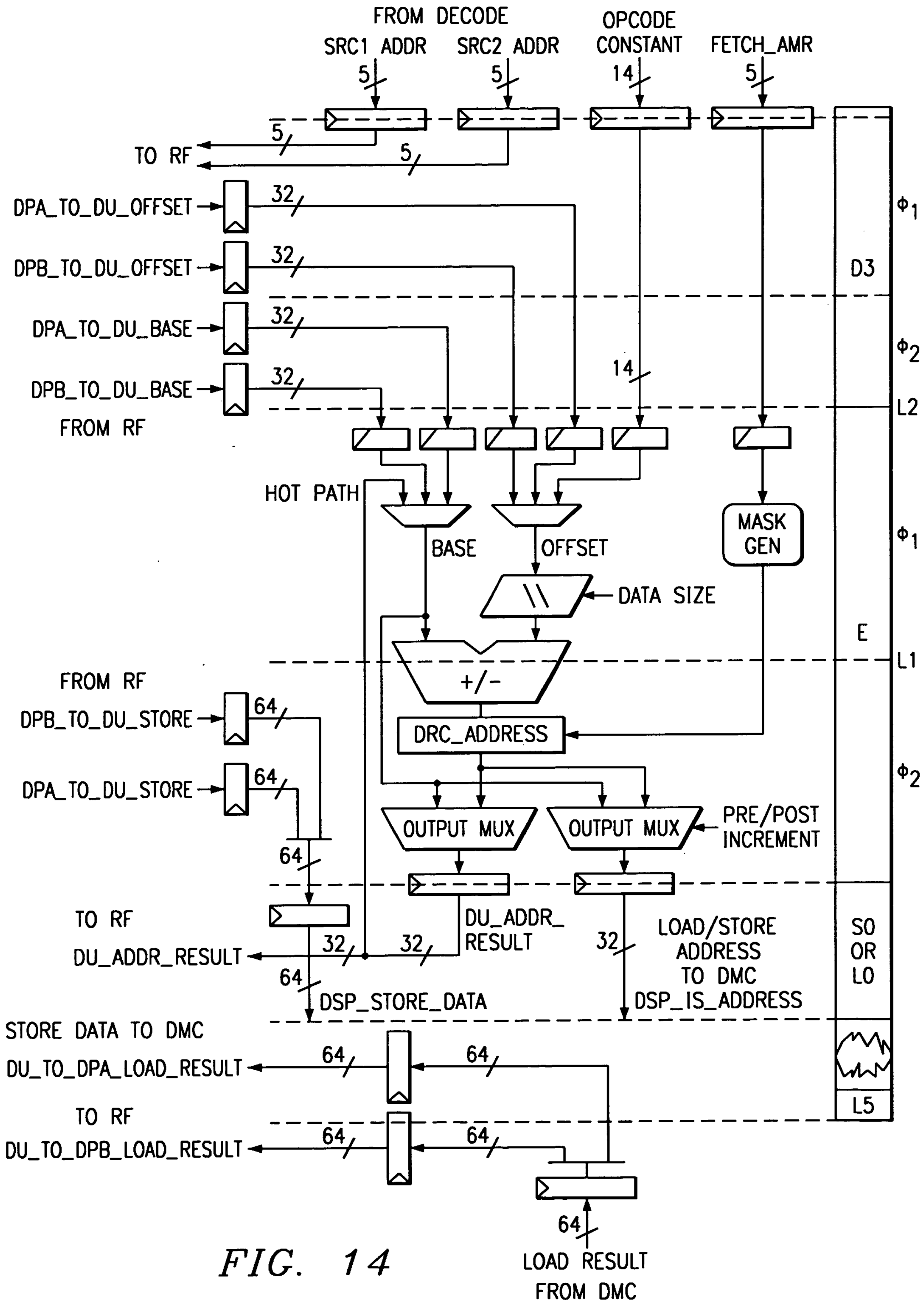

FROM DECODE
SRC1 ADDR
SRC2 ADDR
OPCODE
CONSTANT
FETCH_AMR
5
5
14
5
5
5
TO RF
DPA_TO_DU_OFFSET
DPB_TO_DU_OFFSET
DPA_TO_DU_BASE
DPB_TO_DU_BASE
FROM RF
32
32
32
32
14
Φ1
D3
Φ2
L2
HOT PATH
BASE
OFFSET
MASK GEN
\\
DATA SIZE
Φ1
E
L1
+/-
FROM RF
DPB_TO_DU_STORE
DPA_TO_DU_STORE
64
64
64
DRC_ADDRESS
Φ2
OUTPUT MUX
OUTPUT MUX
PRE/POST INCREMENT
TO RF
DU_ADDR_RESULT
32
32
DU_ADDR_RESULT
32
LOAD/STORE ADDRESS TO DMC
DSP_IS_ADDRESS
64
DSP_STORE_DATA
SO OR LO
STORE DATA TO DMC
DU_TO_DPA_LOAD_RESULT
64
64
TO RF
DU_TO_DPB_LOAD_RESULT
64
64
L5
64
LOAD RESULT FROM DMC

FIG. 14

Mnemonic	Action	Operation
LDB[U] 168	Load byte	Memory B/A XXXXXXba → SSSSSSba (signed (s)) XXXXXXba → 000000ba (unsigned (u))
LDH[U] 170	Load halfword	Memory B/A XXXXdcba → SSSSdcba (s) XXXXdcba → 0000dcba (u)
LDW 172	Load word	Memory B/A hgfedcba → hgfedcba
LDD 174	Load double	Memory B/O A/E ponmlkji hgfedcba → ponmlkji hgfedcba

FIG. 16

Mnemonic	Action	Operation
STB 176	Store byte	B/A Memory XXXXXXba → 000000ba
STH 178	Store halfword	B/A Memory XXXXdcba → 0000dcba
STW 180	Store word	B/A Memory hgfedcba → hgfedcba
STD 182	Store double	B/O A/E Memory ponmlkji hgfedcba → ponmlkji hgfedcba

FIG. 17

Mnemonic	Action	Operation
LDW_BH[U] 184	Word: unpack the bytes into halfwords	Memory B/O A/E hgfedcba → SShgSSfe SSdcSSba (signed (s)) hgfedcba → 00hg00fe 00dc00ba (unsigned (u))
LDW_BHI[U] 186	Word: unpack the bytes into halfwords interleaved	Memory B/O A/E hgfedcba → SShgSSdc SSfeSSba (s) hgfedcba → 00hg00dc 00fe00ba (u)
LDW_HW[U] 188	Word: unpack the halfwords into words	Memory B/O A/E hgfedcba → SSSShgfe SSSSdcba (s) hgfedcba → 0000hgfe 0000dcba (u)
LDD_BH[U] 190	Double: unpack the bytes into halfwords	Memory BO BE AO AE ponmlkji hgfedcba → SSpoSSnm SSlkSSji SShgSSfe SSdcSSba (s) ponmlkji hgfedcba → 00po00nm 00lk00ji 00hg00fe 00dc00ba (u)
LDD_BHI[U] 192	Double: unpack the bytes into halfwords interleaved	Memory BO BE AO AE ponmlkji hgfedcba → SSpoSSlk SSnmSSji SShgSSdc SSfeSSba (s) ponmlkji hgfedcba → 00po00lk 00nm00ji 00hg00dc 00fe00ba (u)
LDD_HW[U] 194	Double: unpack the halfwords into words	Memory BO BE AO AE ponmlkji hgfedcba → SSSSponm SSSSlkji SSSShgfe SSSSdcba (s) ponmlkji hgfedcba → 0000ponm 0000lkji 0000hgfe 0000dcba (u)
LDD_HWI[U] 196	Double: unpack the halfwords into words interleaved	Memory BO BE AO AE ponmlkji hgfedcba → SSSSponm SSSShgfe SSSSlkji SSSSdcba (s) ponmlkji hgfedcba → 0000ponm 0000hgfe 0000lkji 0000dcba (u)

FIG. 18

Mnemonic	Action	Operation
STBH_W 198	Pack the LS byte of each halfword into a word	B/O A/E Memory XXhgXXfe XXdcXXba → hgfedcba
STBHI_W 200	Pack the LS byte of each halfword interleaved into a word	B/O A/E Memory XXhgXXdc XXfeXXba → hgfedcba
STHW_W 202	Pack the LS halfword of each word into a word	B/O A/E Memory XXXXhgfe XXXXdcba → hgfedcba
STBH_D 204	Pack the LS byte of each halfword into a double	BO BE AO AE Memory XXpoXXnm XXlkXXji XXhgXXfe XXdcXXba → ponmlkji hgfedcba
STBHI_D 206	Pack the LS byte of each halfword interleaved into a double	BO BE AO AE Memory XXpoXXlk XXnmXXji XXhgXXdc XXfeXXba → ponmlkji hgfedcba
STHW_D 208	Pack the LS halfword of each word into a double	BO BE AO AE Memory XXXXponm XXXXlkji XXXXhgfe XXXXdcba → ponmlkji hgfedcba
STHWI_D 210	Pack the LS halfword of each word interleaved into a double	BO BE AO AE Memory XXXXponm XXXXhgfe XXXXlkji XXXXdcba → ponmlkji hgfedcba

FIG. 19